© 1994 Balkema, Rotterdam, ISBN 90 5410 389 2

Modern computer technologies in hydrologic data management

N. Papakostas
Division of Computer Engineering, National Technical University of Athens, Greece
I. Nalbantis & D. Koutsoyiannis
Division of Water Resources, Hydraulic and Maritime Engineering, National Technical University of Athens, Greece

ABSTRACT: Advanced water resources management is facilitated by hydrologic data management using modern computer technologies. By means of such technologies, a distributed data management system for hydrologic, meteorologic and hydrogeologic historical information has been built. In this paper, we present the fundamental design and implementation principles and features of the system. Information classification, local and distributed database design and network architecture are described.

INTRODUCTION

During the past three decades, the evolution of computer technologies has changed the way hydrologic and meteorologic data are stored, processed and managed. Several systems that employ such technologies have been built to cover a wide spectrum of methods, tools and requirements. First generation systems, based on central mainframe computers and programming languages like FORTRAN, have pioneered the area of electronic hydrologic and meteorologic data storage and handling. Such systems include NAQUADAT in Canada and WATSTORE in the USA (Rodda & Flanders, 1985). The second generation systems, introducing personal computers and database technologies have emerged during the 1980's. Such systems include HYMOS in the Netherlands (Ogink 1981), HYDATA in Great Britain (Institute of Hydrology, 1991) and HYDRA-PC in Greece (Koutsoyiannis et al., 1991).

Third generation systems, involving relational database technologies, distributed systems and graphics are now being developed. Examples of such systems are (Nalbantis et al. 1992): Sequoia 2000 in the USA (Stonebraker, 1992), National Water Information System II in the USA (USGS, 1991) and CompuMod in Canada (Environment Canada, 1992).

Greece has been mentioned by WMO (1977) as one of the countries that use computer technologies to store hydrometeorologic data. It had lacked, however, a modern, third generation large scale system capable of handling the complex and diverse requirements of the various services that collect and store hydrometeorologic data. Such a system named HYDROSCOPE is now under development.

MAJOR DESIGN GOALS

HYDROSCOPE is created as a joint research project (Tolikas et al., 1993). Fourteen public services participate including ministries, universities and research centres, NTUA being the main partner (Table 1). The project is sponsored by the EC. It aims at the organisation, systemisation and standardisation of hydrologic, meteorologic and hydrogeologic information in Greece. Thus, its primary goal is the creation of a national databank for such information, utilising the capabilities offered by available modern computer technologies. A second goal of major importance is the creation of the framework, within which the participating organisations will be able to co-operatively store, access and process data in a uniform and consistent way. The main principles of the system are autonomy and interchange. The autonomy ensures the independence of each partner in ownership, storage and administration of its own data, according to its specific research and operational needs. The interchange does not apply only to data, but to expertise and know-how as well.

Table 1: HYDROSCOPE participating organisations

Abbre-viation	Organisation
NTUA	National Technical University of Athens (Division of Water Resources, Hydraulic and Maritime Engineering)
NMS	National Meteorological Service
MA	Ministry of Agriculture
MEPPW	Ministry of Environment, Physical Planning and Public Works
NOA	National Observatory of Athens
PPC	Public Power Corporation
AUT/ DHEE	Aristotle University of Thessaloniki (Division of Hydraulic and Environmental Engineering)
AUT/ DE	Aristotle University of Thessaloniki (Division of Energy)
NRCPS "D"	National Research Centre for Physical Sciences "Demokritos"
WSSCA	Water Supply and Sewage Corporation of Athens
UA	University of Athens (Department of Applied Physics)
MIET	Ministry of Industry, Energy and Technology
CRES	Centre for Renewable Energy Sources
HALDLG	Hellenic Agency for Local Development and Local Government

AVAILABLE MODERN COMPUTER TECHNOLOGIES

Several currently available modern computer technologies have been employed, which are capable of supporting both present and future (within reasonable expectations) design goals. They are also commercially available, mature and proven enough not to put at risk the success and viability of the project. Such technologies include:

1. *Relational databases* (Codd, 1970). The state-of-the-art in commercially available database technology offers extensive data handling capabilities and a rich set of support and development tools, well suited for hydrologic applications.

2. *Distributed and decentralised databases*. By distributing computing power and data storage to many locations, as opposed to a huge central monolithic facility, site autonomy and independence, as well as enhanced flexibility and reliability are achieved.

3. *Client-server application architecture*. Data handling and application (front-end) programmes may run on different computers and communicate transparently over a network.

4. *High speed wide-area networks (WAN)*. These are composed of digital dedicated circuits and multi-protocol routers.

5. *Powerful workstations*. Downsizing computing power to the desktop level, while using leading-edge processor, disk and graphics technology results in substantial improvements of the cost/performance ratio.

6. *Easy-to-use graphical user interfaces*. These tools enable easier and friendlier user interaction.

7. *Fourth generation programming languages and object oriented techniques and tools*. Tightly coupled to the data handling facilities, they offer an integrated modern programming environment.

SYSTEM DESCRIPTION AND CURRENT STATE

HYDROSCOPE is a distributed system, currently consisting of 12 nodes. Each node is an Ethernet local area network (LAN) residing at the headquarters of the respective participating organisation (Figure 1). On the LAN, several personal computers (PCs) running MS Windows are attached. These computers are used for data entry and can also run (although slower) HYDROSCOPE's applications. The main component of the LAN is a Hewlett-Packard 9000/s700 UNIX RISC workstation running the INGRES relational distributed database management system and application development tools (INGRES Windows/4GL, etc.), and the HYDROSCOPE applications. The workstation is the database server for the node, holding any data that reside on it, as well as participating in the distributed database. The SQL language (ISO, 1986) is used for data manipulation.

The nodes themselves are interconnected via a wide area network. As most of the nodes are located in the Athens metropolitan area, dedicated leased circuits are used to link them and form a private HYDROSCOPE WAN. A multiprotocol Network Systems router at each node co-ordinates incoming/outgoing traffic for that node. Nodes that reside outside Athens are currently linked to the rest of the network via a public WAN.

The current state of the system is as follows: the WAN is set up and running, although some tests have still to be performed. All node LANs are also complete. Local databases are set up. The distributed database, to which a connection can be made from any node, is also ready. A few components of the database design have still to be implemented. A small set of real data has been inserted in the system for evaluation, debugging, verification and optimisation of both the database and the applications. The application programmes are complete to a great extent, and the system is in its final stages of implementation and testing. However, the majority of data has not been entered into the database; this task has been scheduled for the next phase of the project.

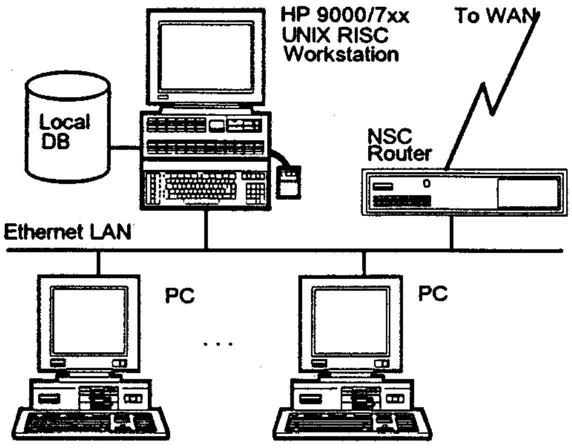

Figure 1: Typical HYDROSCOPE node architecture

CLASSIFICATION OF INFORMATION

There are three main categories of the information stored in the database (Papakostas, 1993). These are:

1. *Application information.* This information has nothing to do with the actual data. It is there just because it is required by the applications. Examples of such information are the topographic maps, the user groups, etc.

2. *Metadata or administrative information* (Figure 2). This is the information about the objects through which the actual data (see below) are located and accessed. These objects are:

Measuring stations. Each station belongs to a single service and may contain several instruments.

Instruments or measuring devices. Each instrument performs a measurement of a single hydrometeorologic variable at various time intervals, e.g. every hour, day, month or even irregularly, and thus "produces" time series data.

Time series characteristics for time series data collected by the instruments. For example, time resolution of the time series, periods with missing values, etc.

Constants, i.e. the characteristics of static data (see below, item 3) about other objects (stations, instruments, time series).

Events about stations, instruments, time series characteristics and constants. Events are chronological information about everything "important" that happens during the object's lifetime, along with an appropriate report. Examples of events are the termination of operation of a station, the repair of an instrument, the infilling of a time series, the creation of a constant, etc.

3. *Actual data* (Figure 2). This is the core of the database, the actual hydrometeorologic information stored. Data can be further classified into two subcategories:

Static (or constant) data are information about stations, instruments and time series that is time invariant or changes infrequently. Examples of such data are the lithological section or drilling information of a hydrogeologic station or the cross-section of a river. They are indexed by the station, instrument or time series to which they refer and are accessed through the constants metadata. No further processing takes place on them.

Time series data are collected from measuring devices or produced by aggregation, processing or transformations on other time series data. They are indexed by the instrument (and date) to which they refer and are accessed through the time series characteristics metadata.

The different types of the Data and Metadata objects are classified into various categories as shown in Figure 2. Stations can be of two types, primary and non-primary. The latter are of low interest as they are used rarely. If a station is non-primary, then the corresponding instruments, time series, constants and events are also considered as non-primary. Instruments are also divided in two categories: real and derived ones (Figure 3). Real instruments are those producing a meaningful measurement for a variable, obtained either by a human observation (e.g. visibility) or by a measuring device (e.g. rainfall, temperature, etc.). Derived instruments do not perform any real measurements; their "measurements" are the result of calculations/transformations on other data. For example, discharge in an hourly or daily time scale corresponds to a derived instrument, as it is derived from a stage-discharge diagram. The latter is also a derived instrument with "measurements" produced from stage and direct discharge measurements (Figure 3).

Time series data can be classified as raw or aggregated (Nalbantis & Tsimbides, 1993). Raw data are the direct or processed measurement outcomes of the instruments, real or derived. Such data form a time series having some time resolution (time step), e.g. hourly, daily, monthly, annual or even irregular. The time resolution is imposed by the frequency of the measurements. Aggregated data are produced when an aggregation function, which effectively broadens the time step (or reduces the time resolution), is applied to raw or even other aggregated data. While raw data may have any time step, aggregated data are stored in the database only in the standard time steps, i.e., hourly, daily, monthly and annual. Other time resolutions may be produced on-line by the user, by applying the aggregation function to the proper data sets.

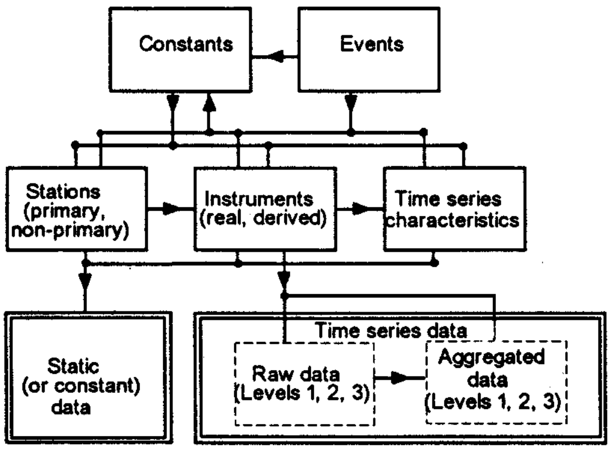
Constants
Events
Stations (primary, non-primary)
Instruments (real, derived)
Time series characteristics
Static (or constant) data
Time series data
Raw data (Levels 1, 2, 3)
Aggregated data (Levels 1, 2, 3)

Figure 2: Data and Metadata objects (in boxes with double and single lines, respectively) and their connections (arrows) and classifications (in brackets)

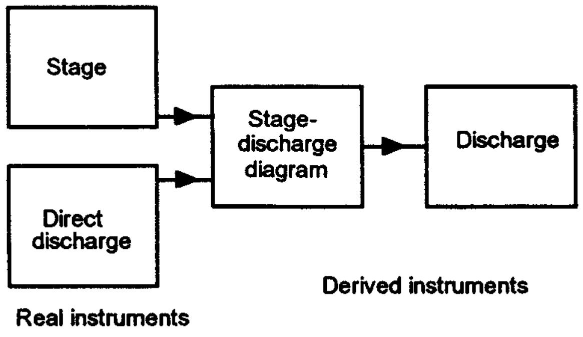

Figure 3: An example of real and derived instruments.

Raw data are subsequently classified in three levels, depending on the type of processing that has been applied to them (Figure 4):

1. Level-1 data are the original data, i.e. the result of the instrument measurements, together with background conditions, i.e. snow, ice, etc.

2. Level-2 data are checked data. The checks are applied to any single data value and include temporal, spatial and internal consistency checks, range checks, etc. Also, corrections to single erroneous values are included. The resulting data are characterised according to their quality and reliability.

3. Level-3 data are the result of infilling operations on time series with missing data, using various techniques and tools, depending on the type of the missing data and the availability of reference data. The techniques used for infilling include: mean value, reciprocal distance, normal ratio (Foufoula, 1983), simple and multiple linear regression (Matalas & Jacobs, 1964; Fiering, 1963), and univariate or multivariate first-order autoregressive models (AR(1); Salas, 1992).

The aggregated data are also classified in the same three levels. The data of a given level can be produced either from raw or aggregated data (with a higher time resolution) of the same level or from aggregated data (with same time resolution) of the previous level (Figure 4).

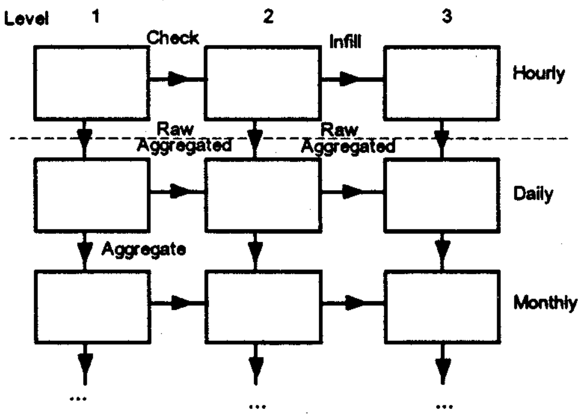

Figure 4: Explanation sketch for the two time series data classifications – raw and aggregated data and their three levels.

GENERAL DATABASE DESIGN

The database design is quite straightforward. Separate tables exist for metadata objects, each row of these tables referring to one object. For example, the stations table contains rows, each row holding the information on one station. Specially formatted integers (HYDROSCOPE id's) uniquely identify these objects. Redundant information on metadata is used to enhance performance whenever possible, to the expense of some disk space. Words instead of plain codes are used for the types of the objects, to enhance user readability.

Each of the participating services maintains its own stations and instruments, and measuring procedures. Every station may also contain various instruments, depending on the service, the variables of interest, the geographical area, etc. Therefore, there is no common measuring standard, format, time step or checking method among the services or even within the same service. Thus, time series data can be identified only by the instrument and the date to which they refer and are stored in the following general standard storage structure:

```
record = instrument_id date status
         <value>
```

where `<value>` is the measured, derived or aggregated value, `instrument_id` is the instrument to which the `<value>` refers, `date` is the respective date, and `status` is the status word.

The status word is a 32-bit (4-byte) integer, whose bits describe the characteristics and the state of the particular record and its `<value>`: the checks and infilling operations that have been applied to it, its level, the background conditions during the measurement (for raw data), etc. It must be noted that, when the `<value>` is acceptable to more than one levels, the same physical database record is used; some status bits are just "turned-on" to indicate that the record belongs to the respective levels.

The `<value>` need not be a single value. In general, it consists of one or more parts, like:

```
<value> = <part0>[<part1>...<partn>]
```

This is because not all instruments measure in the same way. For example, a rainfall measuring instrument measures only the rainfall depth; a wind measuring instrument measures both the wind speed and the wind direction; a soil temperature instrument measures the soil temperature for six different depths. Thus, the corresponding `<value>` field consists of one, two and six parts, respectively. Each part is stored as an integer of a given width (1-, 2- or 4-bytes) and precision, although the real value is a 4-byte floating point number. The integer type has been chosen in order to reduce the disk storage and network bandwidth requirements. Given the measurement unit and the required precision (in decimal digits), the following relationship holds for each part:

$$\texttt{<part>} = \texttt{real\_part\_value} * 10^{\texttt{precision}}$$

To obtain the real value an application has to divide by $10^{\text{precision}}$, and this is the cost of this methodology. For instance, temperatures and rainfall depths are stored as 2-byte integers with a precision of one decimal unit. Since two-byte integers have a maximum value of 32768, a temperature (or rainfall depth) measurement may have a value of up to 32768 / 10 = 3276.8 degrees (or mm). The format used to store the values of a specific time series is stored in the table containing the time series characteristics.

All values are "nullable", i.e. they may have the special "non-value" `null` that is used to denote the absence of a measurement. Thus, for any missing value (e.g. due to temporary instrument malfunction) the corresponding `record` must exist in the time series table, but all (or just the missing) parts are set to `null`. It must be noted however, that because of INGRES restrictions the cost paid to implement this property is high, i.e. one extra byte for each part of the `<value>`.

An alternative storage structure is the list structure. It is appropriate to store a series of `<value>` fields with a constant time step. The list storage structure may be used to store automated instrument readings, e.g. a full day of 24 hourly measurements for rainfall or wind, or a well-defined set of aggregated values, e.g. a full month of 30 or 31 daily values. The general layout of this format is:

```
list_record = instrument_id date status
              num_values time_interval
              <value0> <value1> ...
              value31>
```

for up to 32 values. The `date` field stores the starting date for the list that contains `num_values` elements. The `ith` `<value>` field refers to a date equal to `date + (i-1) • time_interval`. The `<value>` fields in the list storage structure have also formats and parts, i.e. they may actually compose of more than one parts, their width depending on the type of the variable.

Referential integrity rules (Date 1981) have been employed to ensure consistency of the database, e.g. that any instrument belong to one and only one station, etc. Powerful security features are built into the database (Pipili & Papakostas 1992). The scheme used is based on user groups that may or may not be permitted to perform certain operations (select, insert and update - delete) on certain categories of data (raw - aggregated, metadata). The system also permits accounting for any retrieved information. The accounting is based on the type of the data fetched, the number of rows and the processing time of the query.

DISTRIBUTED DATABASE DESIGN

The distributed nature of the system is one of its greatest strengths, although it is a constraint as well. The fact that any data used may (and almost certainly will) be fetched from a remote location greatly affects the design of any application programme. Response time and speed considerations are of major importance. Programmes may not assume that data access can be unlimited. Instead, data access must be carefully co-ordinated and grouped, to avoid unnecessary usage of the limited network bandwidth.

The distributed system offers location transparency. That is, applications and users need not know the exact remote location where requested data are stored. They just issue a query to the distributed database server, which accordingly queries the local databases and fetches any results for the client.

To reduce network usage, several techniques for network bandwidth saving have been used. They greatly reduce the need to use the network and (in most cases) the storage space required. Unfortunately, most of them result in increased application complexity, to handle the extra data structuring.

The techniques used include:

1. *Replication.* Hydrometeorologic data themselves may not be stored at any other location, except for their administrative organisation, but metadata may. So, information about primary stations (non-primary metadata are not replicated), instruments, time series characteristics, events and constants entered to a local database get automatically copied to all other nodes' databases off-line, during the night, when the network is not heavily loaded. Next day, all nodes contain synchronised and consistent information about all system's metadata. Thus, all users have a complete view of the available information on existing data and may decide about their working set by just querying their local database. Using the network is then limited to actually asking for and getting the data. The storage space overhead is minor, compared to the increased speed and flexibility of the method.

2. *Integer values.* As explained previously, the values are stored as integer values. This technique reduces network traffic as opposed to all-four-byte floating point numbers.

3. *List storage structure.* This data storage structure may be used not only to save storage space, but also to transfer over the network smaller amounts of data, which will turn to "unlisted" by the application programme at their destination node.

4. *Linear storage.* This is a variation of the standard storage structure only. (If used with the list storage structure the increased algorithm complexity would not be justified by the gains.) In the linear storage structure, only the end-points, or else the "break-points" of a linearly varying time series are stored. Not stored points are calculated by linear interpolation, resulting in significant savings for some variables (Figure 5). Apparently, the linear storage structure is not of any benefit when the stored values do not follow any linear pattern over a long time period. Typical examples of such a behaviour are most meteorological variables like temperature, air pressure, humidity, etc. However, this is not the case for hydrologic variables like rainfall, river stage or lake stage, which may be stored in this way. The drawback of this method is the increased application complexity.

5. *Remote processing.* A remote processing server is going to be developed. There are some standard hydrologic procedures, such as the extract of the maximum or minimum rainfall depths for a given duration, which may be performed by that server at the node where the data are stored, without transferring any data apart from the query result to the node that requested the information. The result is sent back to the respective server at the requesting node, along with similar results from other nodes. All results are then post-processed and returned to the application. INGRES and SQL use remote processing to some extent (e.g. average or sum calculations) but a more comprehensive approach is required.

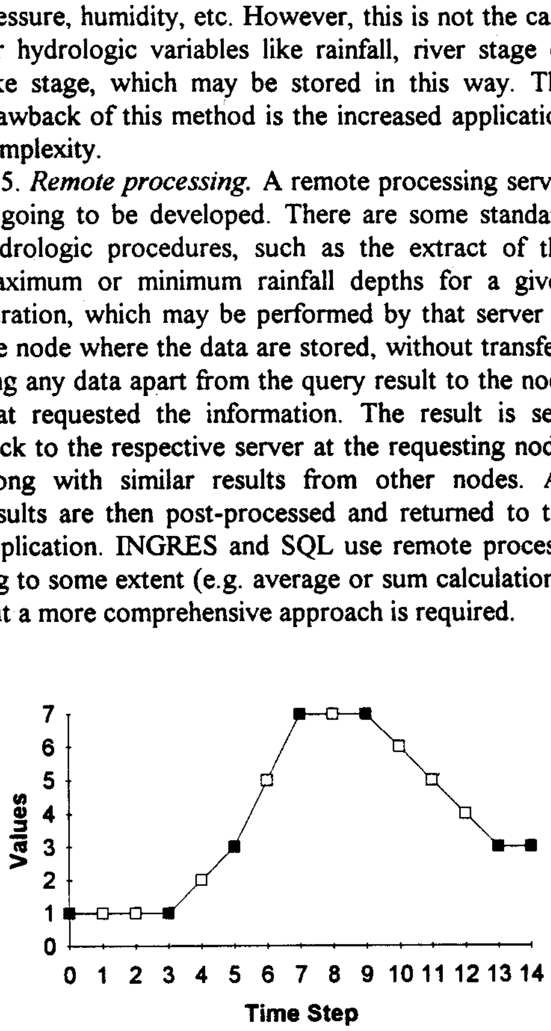

Figure 5: Explanation sketch of linear storage. Filled rectangles represent the records stored using the linear storage technique. The conventional method for the same time series would require the storage of both the filled and the unfilled rectangles – records. Thus, the conventional method of storing this particular time series would require 15 records in the database and eventually over the network, while the linear storage structure requires only 7.

6. *Query limits.* Queries that are going to return too many rows (currently this limit is set to 1000, but

may change according to test results) are rejected by the local database server, before they are executed. This is possible because of the way INGRES statistical query optimiser works. A deferred execution of these queries, e.g. during the night, is planned.

The distributed database and the node local databases have the same schema. Thus, applications need not differentiate their database access according to the particular database used. The distributed database's tables are the logical union of the respective tables at the local nodes' databases. Applications connect as clients to both their node's local database and the distributed database servers. All database access is performed locally, until the user has determined his/her data working set to be queried. Then the query is submitted to the distributed database (Figure 6).

WIDE AREA NETWORK DESIGN

Some of the nodes store no hydrometeorologic data of their own. Some others store few data, while some store the majority of data. It has been estimated that NMS data will account for the 35% of the total data traffic on the WAN, MA data for 20%, MEPPW data for 15%, PPC for 20% and NOA data for 10% of the total data traffic. NTUA also accounts for a lot of traffic of administrative data nature (Papakostas, 1992).

High network bandwidth is of paramount importance, since the speed of the network is the main limiting factor for the design and usage of the system. Other parameters, such as telecommunication and equipment costs, reliability, extendibility, autonomy and automated administration have been taken into account.

It has been finally determined that a backbone design, with multiple fast connections among the main data nodes mentioned above will best satisfy the goals of the system. NTUA and NMS are the major nodes, with the other main nodes connecting to both of them. The rest of the nodes connect to NTUA, because this is the most cost-effective solution (Figure 7).

Nodes not participating in the private WAN currently connect to NTUA via public WAN such as ARIADnet for NRCPS"D" and ITEnet for the Thessaloniki nodes. This choice has been made due to the extremely high cost of such dedicated circuits. High speed multiprotocol routers use these connections to route data from one node to another, using TCP/IP (Socoloftsky & Kale, 1990) over PPP (Perkins & Hobby, 1990) as transport/network/datalink protocols and RIP (Hedrick, 1988), soon to be replaced by OSPF (Moy, 1991), as the routing protocol.

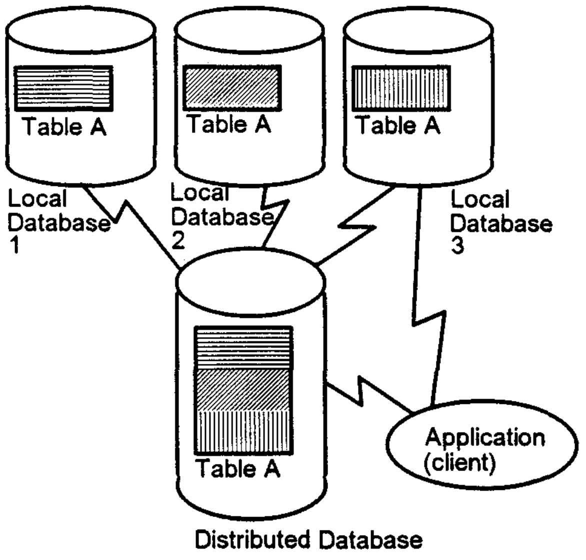

Figure 6: Distributed database architecture. A distributed database table is the local union of all local database tables. In this example only three local databases form the distributed database. HYDROSCOPE's distributed database in composed of five local databases.

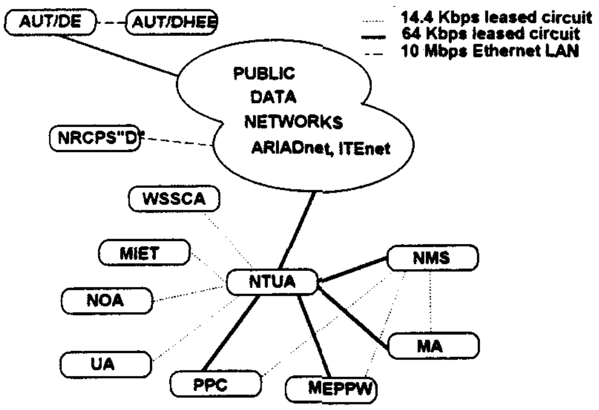

Figure 7: The HYDROSCOPE WAN topology

Despite the careful network design, the obtainable bandwidth is still about three orders of magnitude (Kbps as opposed to Mbps) smaller than local disk-to-memory access. Therefore, the existence of the network is still the main limiting factor of the performance of the system.

APPLICATIONS

HYDROSCOPE applications run as clients of the local and distributed database servers and the network. Clients and servers may reside on different network nodes. In fact, it does not matter for the server on which node the client resides, so that the system is network transparent.

All applications employ a graphical user interface (under X-Windows) to assist user interaction. They communicate to the database server(s) using SQL. Certain time consuming or critical functions are written in C.

ACHIEVEMENTS AND FUTURE DEVELOPMENT

The technological goals of the system design have been met. That is, a network connecting the participating services, local databases on each node storing that node's data, a distributed database that represents the union of the stored local data and proper application programmes have been built. All parts of the system employ modern computer technologies. After its completion the system will be able to store the total amount of hydrometeorologic data in Greece. During the development of the system, extensive inter-service expertise and know-how interchange have formed the basis of strong future collaboration.

The main future development goals of the system include:

1. Completion of the data entry phase, so that the majority of Greek hydrometeorologic data are stored in a common format and are accessed uniformly.
2. Completion of the implementation and debugging phases for the database, the application programmes and the network.
3. Integration of a Geographical Information System, mainly for better management of the application and administrative information.
4. Integration of hydrologic and meteorologic models.
5. Water and air quality data.
6. Satellite and radar data.
7. Data acquisition from telemetric measuring stations and integration of such measurements as raw database data.
8. Possibility for third-party users to connect to HYDROSCOPE and use its services.

ACKNOWLEDGEMENTS

HYDROSCOPE would never be possible without the approval of the project proposal by the Greek General Secretariat of Research and Technology (GSRT), and without the interest and funding by the EC via the STRIDE HELLAS research programme. Additional funding was provided by MIET, MEPPW, MA, WSSCA, NMS, PPC, GSRT and the Ministry of Education.

The system requirements are the result of collective work of many HYDROSCOPE scientists, too numerous to mention here. Without their effort the system design would never be possible. Special thanks are due to D. Tolikas, the project director, T. Sellis, the head of the informatics group, N. Dris, J. Mallas and J. Papageorgiou of NMS for their help on meteorologic database design, S. Ladas and N. Georgiades of AUT/DHEE for their help on hydrogeologic database design, A. Christofides, N. Mamassis and G. Tsakalias of NTUA for their help on hydrologic database design. Also thanks are due to G. Gazepis, D. Kalogeras, P. Papanikolaou, K. Pipili, A. Sakellariou and G. Tsimpides for their help and various useful comments and suggestions during the development of the system.

REFERENCES

Codd, E.F., A relational model of data for large shared data banks, *Communications of the ACM*, 1970.

Date, C., Referential integrity, *Proceedings of the seventh international VLDB conference*, Cannes, France, 1981.

Environment Canada, *CompuMod functional specification*, Report, Environment Canada, Ottawa, Canada, 1991.

Fiering, M.B., Use of correlation to improve estimates of the mean and the variance, *USGS professional paper 434-C*, 1963.

Foufoula-Georgiou, E., Estimation of missing values in monthly rainfall series, *Proceedings in stormwater and water quality model users' group meeting*, EPA-600/9-83-015, 1983.

Hedrick, C.L., Routing Information Protocol, *Internet RFC 1058*, 1988.

Institute of Hydrology, HYDATA: Hydrological database system, Leaflet, Institute of Hydrology, 1991.

ISO, *Database language SQL*, International Standards Organisation, Final Draft ISO 9075-1987 (F), 1986.

Koutsoyiannis, D., Tsolakidis, K. and Mamassis, N., HYDRA-PC: A database system for regional hydrological data management, in G. Tsakiris (ed) *Advances in water resources technology*, Balkema Rotterdam, Netherlands, 1991.

Matalas, N.C. and Jacobs, B., A correlation procedure for augmenting hydrologic data, *USGS professional paper 434-E*, 1964.

Moy, J., Open Shortest Path First, version 2, *Internet RFC 1247*, 1991.

Nalbantis, I. and Tsimbides, G., Data storage levels for raw and processed information and related processing requirements, HYDROSCOPE Technical Report 1/11 (in Greek), NTUA, Athens, Greece, 1993.

Nalbantis, I., Pipili, K. and Tsakalias, G., International experience on archiving and processing of stage, discharge and sediment transport data, HYDROSCOPE Technical Report 1/14 (in Greek), NTUA, Athens, Greece, 1992.

Ogink, H.J.M., HYMOS: A data processing system for hydrological time series, *Proc. International Conference on numerical modelling of river, channel and overland flow for water resources and environmental applications*, Bratislava, Czechoslovakia, 1981.

Papakostas, N., HYDROSCOPE computer systems network study, design and specifications, HYDROSCOPE Technical Report (in Greek), NTUA, Athens, Greece 1992.

Papakostas, N., Database schema design, HYDROSCOPE Technical Report 1/15 (in Greek), NTUA, Athens, Greece, 1993.

Perkins, D. and Hobby R., Point-to-Point Protocol for the transmission of multiprotocol datagrams over point-to-point links, *Internet RFC 1171*, 1990.

Pipili, K. and Papakostas, N., A proposal for the design of the security and accounting system, HYDROSCOPE Technical Report 1/7 (in Greek), NTUA, Athens, Greece, 1992.

Rodda, J.C. and Flanders, A.F., The organisation of hydrological services, Ch. 14, *Facets of Hydrology*, John Wiley & Sons, 1985.

Salas, J.D., Filling in missing observations and extension of records, *Handbook of hydrology*, J.R. Maidment (ed), McGraw-Hill, 1992.

Socoloftsky, T.J. and Kale, C.J., TCP/IP tutorial, *Internet RFC 1180*, 1990.

Stonebraker, M., An overview of the Sequoia 2000 project, Sequoia 2000 Technical Report 91/5, Berkeley, USA 1992.

Tolikas, D., Koutsoyiannis, D. et Xanthopoulos, Th., HYDROSCOPE: Un système d'informations pour l'étude des phénomènes hydroclimatiques en Grèce, *6ème Colloque International de Climatologie*, Thessaloniki, 1993.

USGS (United States Geological Survey), *System requirements specification for the USGS National Water Information System II*, S.B. Mathey (ed), USGS, Reston, USA, 1991.

WMO (World Meteorological Organisation), *Statistical information on activities in operational hydrology*, No 464, Geneva, Switzerland, 1977.